

# WanalQ

## Technical Roadmap for MVP Development

*AI-Powered Civic Engagement Platform for Democratic Participation*

<b>Project Name:</b>	WanalQ: AI-Powered Civic Engagement Platform
<b>Thematic Area:</b>	Generative & Agentic AI
<b>Document Version:</b>	1.5 (Final Submission)
<b>Submission Date:</b>	January 30, 2026
<b>Development Period:</b>	February 2026 - March 2026 (8-Week Sprint)
<b>Development Approach:</b>	Vertical Slice Architecture - Solo Developer
<b>Current Status:</b>	Ready for Incubation Phase

### ■ Key Highlights

- ✓ **Resource-Efficient AI:** Hybrid inference strategy eliminates 24/7 heavy model costs through on-demand serverless architecture
- ✓ **Kenyan Context Expertise:** Custom Swahili NLP models and 500+ curated Kenyan civic documents
- ✓ **Agentic AI Innovation:** Autonomous 'Civic Agent' that routes issues and guides citizens through complex processes
- ✓ **Mobile-First Design:** Progressive Web App optimized for Kenya's mobile-first, bandwidth-constrained market
- ✓ **Proven Velocity:** Vertical slice architecture enables rapid iteration with demo-ready features every week

# 1. Executive Summary

---

This technical roadmap outlines the **expedited 8-week development plan** for WanaIQ's Minimum Viable Product (MVP). Designed for sprint development ending in **March 2026**, this plan leverages high-efficiency tools (React, Node.js, Supabase, Agentic AI) to deliver a production-grade civic engagement solution.

By employing a **Vertical Slice Architecture** as a solo developer assuming multiple functional roles, WanaIQ will deploy fully functional features incrementally, ensuring a **demo-ready product at every milestone**. This approach maximizes development velocity while maintaining code quality and user experience standards.

## 1.1 MVP Core Objectives

Objective	Description	Success Metric
■ Agentic AI Assistant	Deploy 'Civic Agent' that intelligently routes citizen issues using LLM-driven classification and multi-step workflow execution	<ul style="list-style-type: none"><li>• 90%+ routing accuracy</li><li>• &lt;2s response time</li><li>• 200+ queries/day</li></ul>
■ RAG Pipeline ("Civic Brain")	Enable citizens to query Kenyan Constitution and laws in natural language with verified, cited responses.	<ul style="list-style-type: none"><li>• 85%+ answer accuracy</li><li>• 100% source citation</li><li>• 10+ languages supported</li></ul>
■ Content Analysis AI	Automated video transcription, sentiment analysis, and metadata extraction for CivicClips educational content	<ul style="list-style-type: none"><li>• &lt;2 min processing time</li><li>• 95%+ transcription accuracy</li><li>• Auto-moderation active</li></ul>
■ Cloud-Native Scalability	Serverless architecture using Next.js, Node.js, and Supabase for high performance and auto-scaling.	<ul style="list-style-type: none"><li>• &lt;200ms API latency (p95)</li><li>• 99.9%+ uptime</li><li>• 1000+ concurrent users</li></ul>
■ Multilingual Support	Real-time translation supporting English, Swahili, and 10+ Kenyan languages for inclusive civic access	<ul style="list-style-type: none"><li>• 10+ languages live</li><li>• &lt;500ms translation time</li><li>• 80%+ quality score</li></ul>

## 2. Technical Architecture Overview

---

### 2.1 Technology Stack

WanalQ employs a modern, production-grade technology stack optimized for rapid development, resource efficiency, and scalability. Each technology choice is justified by specific technical requirements:

Component	Technology	Purpose	Justification
Frontend Web	React 18 + TypeScript Next.js 14	Responsive web with SSR/SSG	SEO optimization, fast load, type safety
Frontend Mobile	React Native + Expo	Cross-platform iOS/Android	Code reuse 70%+, rapid iteration
Backend API	Python/Node.js + Express GraphQL	RESTful and GraphQL APIs	Async I/O for AI calls, reduces over-fetching
Database (Primary)	Supabase PostgreSQL + pgvector	Structured data + vectors	Built-in auth, real-time, vector search
Database (Content)	MongoDB Atlas	Unstructured content	Flexible schema, excellent performance
AI/ML Core	Llama 3 via Groq API	LLM inference	Sub-100ms latency, 10x faster, cost-effective
Vector Database	Supabase pgvector	RAG embeddings	Native PostgreSQL, no separate service
NLP Processing	spaCy + Custom models	Swahili/English NLP	Production-ready, custom Kenyan training
Computer Vision	OpenCV + Python	Image verification	Microservice, on-demand scaling
Transcription	Whisper API	Video transcription	State-of-art accuracy, multi-language
Caching	Redis Cloud	Session + response cache	70% cost reduction, sub-ms latency
Cloud Infrastructure	AWS Lambda + EC2 + S3 / Supabase + S3	Serverless + hosting	Pay-per-use, auto-scaling, global CDN
CDN	Cloudflare	Content delivery	DDoS protection, edge caching, analytics
Authentication	OAuth 2.0 + JWT	Secure auth	Industry standard, social login, stateless
CI/CD	Vercel + GitHub Actions	Automated deployment	Zero-config deploys, auto previews, rollback

## 2.2 System Architecture

WanalQ follows a **microservices architecture** with clear separation of concerns, enabling independent scaling and deployment of each component:

Layer	Components	Responsibilities	Scaling Strategy
1. Presentation Layer	<ul style="list-style-type: none"><li>• Next.js web app</li><li>• React Native mobile</li><li>• PWA shell</li></ul>	<ul style="list-style-type: none"><li>• Server-side rendering</li><li>• Static generation</li><li>• Client-side hydration</li><li>• Responsive design</li></ul>	Edge caching via CDN, static asset optimization, code splitting
2. API Gateway Layer	<ul style="list-style-type: none"><li>• Express.js REST API</li><li>• GraphQL endpoint</li><li>• WebSocket server</li></ul>	<ul style="list-style-type: none"><li>• Request routing</li><li>• Authentication/authorization</li><li>• Rate limiting</li><li>• Response caching</li></ul>	Horizontal scaling via load balancer, stateless design
3. Application Logic Layer	<ul style="list-style-type: none"><li>• Business logic</li><li>• Data validation</li><li>• Workflow orchestration</li></ul>	<ul style="list-style-type: none"><li>• User management</li><li>• Content moderation</li><li>• Notification system</li><li>• Analytics processing</li></ul>	Containerized services, auto-scaling based on CPU/memory
4. AI Services Layer	<ul style="list-style-type: none"><li>• Civic Agent Router</li><li>• RAG Engine</li><li>• Video Analyzer</li><li>• Translation service</li></ul>	<ul style="list-style-type: none"><li>• LLM inference</li><li>• Vector search</li><li>• CV processing</li><li>• NLP analysis</li></ul>	Serverless functions, on-demand execution, cold start optimization
5. Data Layer	<ul style="list-style-type: none"><li>• PostgreSQL</li><li>• MongoDB</li><li>• Redis</li><li>• S3 storage</li></ul>	<ul style="list-style-type: none"><li>• Persistent storage</li><li>• Document storage</li><li>• Cache management</li><li>• Media storage</li></ul>	Database replication, read replicas, sharding ready

### 3. Development Phases & Milestones (8-Week Sprint)

The 8-week development sprint is organized into three progressive phases, with each milestone delivering testable, demo-ready functionality:

Milestone	Timeline	Deliverables	Success Criteria	Testing
M1.1 Infrastructure Setup	Feb Week 1	<ul style="list-style-type: none"> <li>• Git linked to portal</li> <li>• Supabase configured</li> <li>• CI/CD pipeline</li> <li>• Development environment</li> <li>• Staging deployment</li> </ul>	<ul style="list-style-type: none"> <li>✓ Repo visible in portal</li> <li>✓ Database accepts connections</li> <li>✓ Auto-deploy on commit</li> <li>✓ 100% env documented</li> </ul>	<ul style="list-style-type: none"> <li>• Connection tests</li> <li>• Pipeline smoke tests</li> <li>• Access control verification</li> </ul>
M1.2 API Migration	Feb Week 1-2	<ul style="list-style-type: none"> <li>• Node.js Express server</li> <li>• GraphQL schema</li> <li>• Authentication middleware</li> <li>• Rate limiting</li> <li>• API documentation</li> </ul>	<ul style="list-style-type: none"> <li>✓ All endpoints responding</li> <li>✓ Auth flow working</li> <li>✓ 100 req/min limit enforced</li> <li>✓ Swagger docs live</li> </ul>	<ul style="list-style-type: none"> <li>• Unit tests (80% coverage)</li> <li>• Integration tests</li> <li>• Load testing (100 users)</li> </ul>
M2.1 Civic Agent Router	Feb Week 3	<ul style="list-style-type: none"> <li>• LLM classification logic</li> <li>• Department routing</li> <li>• Intent recognition</li> <li>• Multi-turn conversation</li> <li>• Confidence scoring</li> </ul>	<ul style="list-style-type: none"> <li>✓ 90%+ routing accuracy</li> <li>✓ &lt;2s response time</li> <li>✓ 5+ issue categories</li> <li>✓ Conversation state persists</li> </ul>	<ul style="list-style-type: none"> <li>• 100 test cases</li> <li>• Accuracy benchmarking</li> <li>• Latency profiling</li> <li>• Edge case testing</li> </ul>
M2.2 RAG Pipeline	Feb Week 4	<ul style="list-style-type: none"> <li>• Document ingestion (500+ docs)</li> <li>• Vector embeddings</li> <li>• Similarity search API</li> <li>• Context injection</li> <li>• Source citations</li> </ul>	<ul style="list-style-type: none"> <li>✓ 85%+ answer accuracy</li> <li>✓ 100% responses cited</li> <li>✓ &lt;1s search latency</li> <li>✓ No hallucinations detected</li> </ul>	<ul style="list-style-type: none"> <li>• 50 civic Q&amp;A pairs</li> <li>• Citation validation</li> <li>• Hallucination detection</li> <li>• Performance testing</li> </ul>
M2.3 CivicClips AI	Mar Week 1	<ul style="list-style-type: none"> <li>• Video upload pipeline</li> <li>• Whisper transcription</li> <li>• Sentiment analysis</li> <li>• Auto-captioning</li> <li>• Content moderation</li> </ul>	<ul style="list-style-type: none"> <li>✓ &lt;2 min processing</li> <li>✓ 95%+ transcription accuracy</li> <li>✓ Sentiment scored</li> <li>✓ Inappropriate content flagged</li> </ul>	<ul style="list-style-type: none"> <li>• 20 video samples</li> <li>• Accuracy validation</li> <li>• Processing time tests</li> <li>• Moderation precision</li> </ul>
M3.1 Gamification System	Mar Week 2	<ul style="list-style-type: none"> <li>• Civic points logic</li> <li>• User reputation scores</li> <li>• Achievement badges</li> <li>• Leaderboards</li> <li>• Quest system</li> </ul>	<ul style="list-style-type: none"> <li>✓ Points awarded correctly</li> <li>✓ Leaderboard updates real-time</li> <li>✓ 10+ achievements</li> <li>✓ Engagement +50%</li> </ul>	<ul style="list-style-type: none"> <li>• Points calculation tests</li> <li>• Leaderboard accuracy</li> <li>• Performance under load</li> <li>• User acceptance testing</li> </ul>
M3.2 Mobile Optimization	Mar Week 2-3	<ul style="list-style-type: none"> <li>• Progressive Web App</li> <li>• Offline capability</li> <li>• Push notifications</li> <li>• Touch gestures</li> <li>• App manifest</li> </ul>	<ul style="list-style-type: none"> <li>✓ Lighthouse score &gt;90</li> <li>✓ Offline mode functional</li> <li>✓ Notifications working</li> <li>✓ iOS/Android tested</li> </ul>	<ul style="list-style-type: none"> <li>• Device compatibility</li> <li>• Offline scenario tests</li> <li>• Performance audit</li> <li>• User experience testing</li> </ul>
M3.3 Final Launch	Mar Week 3-4	<ul style="list-style-type: none"> <li>• Production deployment</li> <li>• SSL certificates</li> <li>• Monitoring setup</li> <li>• Demo video</li> <li>• Documentation</li> </ul>	<ul style="list-style-type: none"> <li>✓ Public URL live</li> <li>✓ 99.9%+ uptime</li> <li>✓ Demo video &lt;5 min</li> <li>✓ All docs complete</li> </ul>	<ul style="list-style-type: none"> <li>• Full regression testing</li> <li>• Security audit</li> <li>• Performance benchmarking</li> <li>• User acceptance testing</li> </ul>

## 4. Development Strategy: Vertical Slice Architecture

As a **solo developer** participating in this hackathon, the project employs a Vertical Slice Architecture where I assume multiple 'Functional Hats' to execute the roadmap efficiently. This approach, augmented by AI tools and modern frameworks, enables single-developer velocity comparable to a small team:

Functional Hat	Core Responsibilities	Efficiency Tools ("Virtual Team")	Time Allocation
■■■ Full Stack Lead	<ul style="list-style-type: none"><li>• React/Node.js development</li><li>• Database schema design</li><li>• API security implementation</li><li>• Code review and refactoring</li></ul>	Supabase: Instant backend (auth, DB, real-time) Vercel: Zero-config CI/CD shadcn/ui: Pre-built components Copilot: Code completion	40% of time (~16 hrs/week)
■ AI/ML Engineer	<ul style="list-style-type: none"><li>• RAG pipeline construction</li><li>• LLM prompt engineering</li><li>• Agentic workflow logic</li><li>• Model evaluation</li></ul>	LangChain: Agent orchestration HuggingFace: Pre-trained models Groq API: Fast LLM inference Pinecone: Vector database	30% of time (~12 hrs/week)
■ Product & Design	<ul style="list-style-type: none"><li>• UI/UX implementation</li><li>• User story mapping</li><li>• Feature prioritization</li><li>• Demo creation</li></ul>	Tailwind CSS: Rapid styling v0.dev: AI UI generation Figma: Design mockups Loom: Demo videos	20% of time (~8 hrs/week)
■ DevOps & QA	<ul style="list-style-type: none"><li>• Infrastructure setup</li><li>• Monitoring configuration</li><li>• Test automation</li><li>• Performance optimization</li></ul>	GitHub Actions: CI/CD Playwright: E2E testing Datadog: Monitoring Sentry: Error tracking	10% of time (~4 hrs/week)

### Key Productivity Multipliers:

- **AI-Assisted Development:** GitHub Copilot, Claude, and ChatGPT reduce coding time by 40-50% for boilerplate and standard patterns
- **Managed Services:** Supabase, Vercel, and MongoDB Atlas eliminate infrastructure management overhead
- **Component Libraries:** shadcn/ui and Tailwind provide production-ready components, avoiding custom CSS
- **Testing Automation:** Playwright and Jest enable continuous testing without manual QA efforts
- **Low-Code Tools:** v0.dev generates initial UI layouts, reducing design-to-code translation time by 60%

## 5. AI/ML Technical Approach (Resource-Efficient Strategy)

---

### 5.1 Hybrid Inference Architecture

To avoid the 'resource-heavy' trap of running large AI models 24/7, WanaIQ implements a strategic **hybrid inference approach** that separates heavy and light workloads:

Workload Type	Processing Method	Infrastructure	Cost Model	Latency
■ Heavy Lifting (Video/Image)	On-Demand Serverless Functions	AWS Lambda with Python runtime (2GB memory)	Pay-per-invocation \$0.20/100 videos vs \$50/mo always-on	30-120s (acceptable for async tasks)
■ Agent Logic (Text/Chat)	API-Based Inference (Groq/Replicate)	Managed API endpoints with global CDN	Pay-per-token \$0.10/1M tokens vs \$200/mo GPU	<100ms (critical for real-time chat)
■ Vector Search (RAG)	Managed Vector DB (Supabase pgvector)	PostgreSQL extension with indexing	Included in DB costs \$25/mo database	<50ms (optimized with indexes)
■ Batch Processing (Analytics)	Scheduled Jobs (Cron + Lambda)	Serverless functions run off-peak hours	\$0.50/day for daily jobs	Not time-critical (overnight OK)

### 5.2 RAG over Fine-Tuning: Strategic Decision

Factor	RAG Approach (WanaIQ Choice)	Fine-Tuning Approach	Decision
Development Time	1-2 weeks to implement with existing documents	4-6 weeks for data prep, training, and validation	✓ RAG 3x faster
Infrastructure Cost	\$0/month (API + vector DB)	\$500-1000/month (GPU training + serving)	✓ RAG is cheaper
Update Flexibility	Add new documents in minutes, instant availability	Retrain model (days), validate, redeploy	✓ RAG real-time updates
Accuracy & Verification	Responses grounded in actual docs, 100% source citations	Model may hallucinate, difficult to verify sources	✓ RAG verifiable
Hackathon Suitability	MVP ready in 1 week, demo-friendly	Requires weeks before usable results	✓ RAG sprint-compatible

**Conclusion:** RAG provides superior accuracy, flexibility, and development speed. Fine-tuning requires expensive hardware. RAG (Retrieval-Augmented Generation) costs virtually nothing on our stack because we store embeddings in our existing PostgreSQL database and inject context into the LLM prompt. This is not just cheaper, it's more accurate for legal documents.

## 5.3 Detailed AI Component Specifications

Component	Technical Specification	Performance Target	Fallback Strategy
Civic Agent Router	<ul style="list-style-type: none"> <li>Model: Llama 38B via Groq</li> <li>Context: 8k tokens</li> <li>Temperature: 0.3</li> <li>Few-shot: 5 examples/category</li> </ul>	<ul style="list-style-type: none"> <li>Accuracy: &gt;90%</li> <li>Latency: &lt;2s</li> <li>Throughput: 100 req/min</li> </ul>	Fallback to rule-based classifier if API down; alert admin
RAG Engine	<ul style="list-style-type: none"> <li>Embeddings: sentence-transformers</li> <li>Vector DB: pgvector</li> <li>Chunk size: 512 tokens</li> <li>Top-k retrieval: 5 chunks</li> </ul>	<ul style="list-style-type: none"> <li>Answer accuracy: &gt;85%</li> <li>Search latency: &lt;500ms</li> <li>Citation rate: 100%</li> </ul>	Serve cached responses; queue queries for batch processing
Video Transcription	<ul style="list-style-type: none"> <li>Model: Whisper Large v3</li> <li>Languages: EN, SW, 10+</li> <li>Format: SRT captions</li> <li>Post-process: Punctuation, capitalization</li> </ul>	<ul style="list-style-type: none"> <li>Accuracy: &gt;95% WER</li> <li>Processing: &lt;2 min/video</li> <li>Supported: 15+ languages</li> </ul>	Use Whisper Medium for faster processing; manual correction UI
Sentiment Analysis	<ul style="list-style-type: none"> <li>Model: Custom spaCy pipeline</li> <li>Training: Kenyan social media corpus</li> <li>Classes: Positive, Negative, Neutral</li> <li>Confidence threshold: 0.7</li> </ul>	<ul style="list-style-type: none"> <li>F1 score: &gt;0.80</li> <li>Latency: &lt;100ms</li> <li>Batch: 1000 posts/sec</li> </ul>	Generic sentiment model if custom unavailable; manual review flag
Translation	<ul style="list-style-type: none"> <li>Service: Google Translate API</li> <li>Languages: 15+ Kenyan languages</li> <li>Caching: Redis (24h TTL)</li> <li>Quality: Human validation sample</li> </ul>	<ul style="list-style-type: none"> <li>Cache hit rate: &gt;70%</li> <li>Latency: &lt;300ms</li> <li>Cost: &lt;\$10/day</li> </ul>	Show original with translation disclaimer; community corrections

## 6. Risk Management & Mitigation Strategies

Risk Category	Impact Level	Potential Impact	Mitigation Strategy	Contingency Plan
■■ Technical: Integration Complexity	MEDIUM	Node.js + Python services fail to communicate, blocking AI features	<ul style="list-style-type: none"> <li>Strict API contracts (OpenAPI)</li> <li>Redis message queues</li> <li>Comprehensive integration tests</li> <li>Mock services for development</li> </ul>	<ul style="list-style-type: none"> <li>Fallback to monolithic architecture temporarily</li> <li>Manual data transfer</li> <li>Extend timeline by 1 week</li> </ul>
■ AI/ML: Model Hallucinations	HIGH	AI provides incorrect legal information, damaging user trust and platform credibility	<ul style="list-style-type: none"> <li>Strict RAG grounding prompts</li> <li>Response validation layer</li> <li>Confidence thresholds (&gt;0.8)</li> <li>Human review for low-confidence</li> <li>Explicit disclaimers</li> </ul>	<ul style="list-style-type: none"> <li>Disable AI chat temporarily</li> <li>Switch to curated FAQ</li> <li>Community fact-checking</li> <li>Legal expert hotline</li> </ul>
■ Security: Data Privacy Breach	CRITICAL	User data exposed, legal liability, reputational damage, user exodus	<ul style="list-style-type: none"> <li>Encryption at rest and transit</li> <li>Regular security audits</li> <li>Penetration testing (week 6)</li> <li>Minimal data collection</li> <li>Access controls + audit logs</li> </ul>	<ul style="list-style-type: none"> <li>Incident response plan</li> <li>Immediate user notification</li> <li>Data breach insurance</li> <li>Legal compliance team</li> <li>Temporary shutdown if needed</li> </ul>
■ Operational: API Rate Limits	MEDIUM	LLM API throttling during demos, poor user experience, high costs	<ul style="list-style-type: none"> <li>Request queuing system</li> <li>Load balancing across providers</li> <li>Aggressive caching (Redis)</li> <li>Rate limit monitoring</li> <li>Budget alerts (\$100/day cap)</li> </ul>	<ul style="list-style-type: none"> <li>Fallback to lighter models</li> <li>Queue non-urgent requests</li> <li>Upgrade API tier</li> <li>Implement waiting room</li> </ul>
■ Content: Multilingual Support	MEDIUM	Poor Swahili translation quality, alienates majority users	<ul style="list-style-type: none"> <li>Custom spaCy models for Swahili</li> <li>Community validation program</li> <li>Human translators for critical content</li> <li>Progressive language expansion</li> <li>User feedback loops</li> </ul>	<ul style="list-style-type: none"> <li>English-only MVP first</li> <li>Community-driven translations</li> <li>Partner with translation NGOs</li> <li>Delay language launch by 2 weeks</li> </ul>
■ Timeline: Scope Creep	HIGH	Feature additions delay MVP launch, miss March deadline	<ul style="list-style-type: none"> <li>Strict milestone-based development</li> <li>Weekly scope reviews with mentor</li> <li>Feature flags for optional items</li> <li>"Must have" vs "Nice to have" matrix</li> <li>Time-boxed sprints</li> </ul>	<ul style="list-style-type: none"> <li>Cut gamification (M3.1)</li> <li>Reduce language support to 3</li> <li>Skip mobile optimization</li> <li>Focus on core AI features only</li> </ul>

## 7. Success Metrics & Key Performance Indicators

WanalQ's success will be measured across four dimensions: **technical performance**, **user engagement**, **AI effectiveness**, and **civic impact**. These metrics guide development priorities and validate our approach:

Category	Metric	MVP Target	Measurement Method	Review Frequency
■ Technical Performance	API Response Time (p95)	< 200ms	Datadog APM monitoring	Real-time dashboard
	Page Load Time (LCP)	< 2.5s	Lighthouse CI, Web Vitals	Every deployment
	System Uptime	> 99.9%	UptimeRobot, Pingdom	Daily report
	Error Rate	< 0.1%	Sentry error tracking	Real-time alerts
	Test Coverage	> 80%	Jest coverage reports	Every commit (CI)
■ AI Effectiveness	Agent Routing Accuracy	> 90%	Ground truth validation (100 cases)	Weekly evaluation
	RAG Answer Accuracy	> 85%	Human evaluation (50 Q&A pairs)	Weekly evaluation
	Hallucination Rate	< 5%	Manual review + user reports	Daily sampling (20 responses)
	AI Cache Hit Rate	> 70%	Redis analytics	Daily report
	Transcription Accuracy (WER)	< 5%	Manual validation (10 videos/week)	Weekly evaluation
■ User Engagement	Monthly Active Users	500+	Mixpanel analytics	Weekly cohort analysis
	Daily Active Users	100+	Session tracking	Daily dashboard
	Avg. Session Duration	> 5 minutes	Google Analytics	Weekly report
	7-Day User Retention	> 30%	Cohort retention analysis	Weekly cohort tracking
	AI Assistant Usage	200+ queries/day	Database query logs	Daily metrics
■■ Civic Impact	Civic Reports Submitted	50+	Action Hub database	Weekly tally
	CivicClips Views	1000+	Video analytics	Weekly report
	Promise Tracking Votes	500+	Voting system metrics	Weekly tally
	Languages Supported	10+	Translation system config	Milestone-based
	User Satisfaction (NPS)	> 40	In-app surveys (monthly)	Monthly survey

## 8. Visual Development Timeline (Gantt Chart)

Phase/Milestone	Feb W1	Feb W2	Feb W3	Feb W4	Mar W1	Mar W2	Mar W3	Mar W4
<b>PHASE 1: Foundation</b>	■■■	■■■						
M1.1: Infrastructure	■■■							
M1.2: API Migration	■■■	■■■						
<b>PHASE 2: AI Core</b>			■■■	■■■	■■■			
M2.1: Civic Agent			■■■					
M2.2: RAG Pipeline				■■■				
M2.3: CivicClips AI					■■■			
<b>PHASE 3: Production</b>						■■■	■■■	■■■
M3.1: Gamification						■■■		
M3.2: Mobile PWA						■■■	■■■	
M3.3: Launch & Demo							■■■	■■■
<b>CONTINUOUS ACTIVITIES</b>								
Testing & QA	■■	■■	■■	■■	■■	■■	■■	■■
Documentation	■■	■■	■■	■■	■■	■■	■■	■■
Mentor Check-ins	✓	✓	✓	✓	✓	✓	✓	✓

**Legend:** ■■■ = Active development | ■■ = Continuous activity | ✓ = Scheduled checkpoint

## 9. Conclusion & Commitment to Excellence

---

This technical roadmap provides a comprehensive, actionable blueprint for delivering WanaIQ's MVP within the **8-week sprint timeline ending in March 2026**. Our approach balances technical innovation with pragmatic resource management, ensuring we build a functional, demonstrable platform that showcases the transformative potential of Generative and Agentic AI in civic engagement.

### ■ Key Differentiators

- ✓ **Resource-Efficient AI:** Hybrid inference strategy eliminates 24/7 heavy model costs through serverless architecture, reducing operational costs by 90% while maintaining performance
- ✓ **Kenyan Context Expertise:** Custom Swahili NLP models and 500+ curated civic documents ensure culturally relevant, accurate responses that generic AI platforms cannot match
- ✓ **Mobile-First Design:** Progressive Web App optimized for Kenya's mobile-first, bandwidth-constrained market with offline capabilities and sub-2s load times
- ✓ **Agentic AI Innovation:** Goes beyond simple chatbots to implement autonomous agents that actively guide citizens through multi-step civic processes, demonstrating true agentic capabilities
- ✓ **Proven Solo Velocity:** Vertical slice architecture combined with AI-assisted development tools enables single-developer productivity comparable to a 3-4 person team

### ■ Success Validation

Each milestone includes **quantifiable success criteria** measured through automated testing, human evaluation, and user feedback. Weekly progress updates will demonstrate milestone completion with working demos, ensuring continuous validation of our approach.

### ■ Mentor Engagement

The WanaIQ development process is **committed to transparent execution**. Our Git repository is linked to the hackathon portal providing real-time visibility into development progress. We welcome mentor guidance throughout the incubation journey and commit to:

- Weekly progress updates with working demo videos showcasing completed milestones
- Bi-weekly live demo sessions for feedback and course correction
- Real-time code visibility through public GitHub repository with detailed commit messages
- Documented decision rationale for all major technical choices
- Proactive communication of blockers or timeline risks within 24 hours of identification

### ■ Vision Beyond MVP

While this roadmap focuses on MVP delivery, WanaIQ's architecture is designed for national scale from day one. Post-hackathon growth will expand to all 47 Kenyan counties, integrate with official government systems, and serve as a template for civic engagement platforms across Africa. We are not just building an app, we are **engineering a more transparent, responsive, and resilient democracy.**

---

**Prepared by:** WanaIQ Development Team (Nahashon Mwangi)

**Submission Date:** January 30, 2026

**Contact:** Via Hackathon Portal

**Document Version:** 1.5 (Final Submission)

---